Wrocław, 3 grudnia 2023

Podstawy Programowania

Laboratorium 5

Operacje na plikach

prowadzący: *mgr inż. Marta Lampasiak, mgr inż. Michał Jaroszczuk*

1 Wprowadzenie

Celem zajęć jest poznanie podstaw programowania w języku `C/C++`, w szczególności oprogramowanie prostej bazy danych wykorzystującej reprezentacje w postaci tablicy struktur lub tablicy wskaźników na struktury. Rozbudowanie programu o operacje archiwizacji danych w pamięci zewnętrznej w postaci plików tekstowych lub binarnych.

Zapis lub odczyt danych może zrealizować przy użyciu biblioteki `fstream`. Po załączeniu wspomnianej biblioteki pierwszym krokiem jest utworzenie uchwytu do pliku poprzez utworzenie obiektu klasy `fstream`. Domyśle utworzona zmienna nie wskazuję na żaden plik. Poprzez użycie metody `open(const char * path, mode)` można otworzyć plik do odczytu lub zapisu. Jako ścieżkę (*path*) można podać ścieżkę względną lub bezwzględną do pliku. Następnie należy wskazać w jakim trybie chcemy otworzyć plik.

`std::fstream::in`	zezwolenie na odczytywanie danych z pliku.
`std::fstream::out`	zezwolenie na zapisywanie danych do pliku.
`std::fstream::app`	zezwolenie na zapisywanie danych do pliku, ale dane mogą być zapisywane tylko i wyłącznie na końcu pliku.

```cpp
#include <iostream>
#include <fstream>

int main()
{
  std::fstream file;
  file.open("data.txt", std::fstream::in);
  if (file.is_open())
  {
    int x;
    file >> x;
    std::cout << x << " ";
    while (!file.eof()) {
      file >> x;
      std::cout << x << " ";
    }
    std::cout << "\n";
    file.close();
  }
}
```

Dane z pliku można odczytać przy użyciu strumienia `plik>>zmienna;` oraz analogicznie zapisać `plik<<zmienna;`. Wewnątrz klasy `fstream` jest wiele przydatnych metod takich jak: `eof()` do sprawdzenia czy wskaźnik pliku znajduje się na jego końcu oraz `is_open()` do sprawdzenia czy udało się poprawnie otworzyć plik. Przy użyciu strumienia z pliku odczytujemy kolejne dane oddzielone białymi znakami:

```
6 3 2
1 7
```

bez znaczenia czy jest to spacja, tabulacja czy nowa linia.

Struktury definiujemy przy użyciu słowa kluczowego `struct`. Struktury służą do przechowywania złożonych danych, mogą posiadać składowe różnego typu. Na końcu definicji struktury zawsze należy umieścić średnik. Analogicznie jak przy zwykłych zmiennych również można tworzyć tablice struktur. Wczytanie oraz zapis wygląda bardzo podobnie, jednak każdą składową należy wczytać osobno.

```
#include <iostream>
#include <fstream>

struct point
{
  double x, y;
};

void print(point P) {
  std::cout << "(" << P.x << "," << P.y << ")\n";
}

void rnd(point tab[], int n) {
  for (int i=0; i < n; i++){
    tab[i].x = (float)(rand() % 10 - 5)/5;
    tab[i].y = (float)(rand() % 10 - 5)/5;
  }
}

std::string save(point tab[], int n, std::string path) {
  std::fstream file;
  file.open(path, std::fstream::out);
  file << n <<"\n";
  for (int i = 0; i < n; i++) {
    file << tab[i].x << " " << tab[i].y <<"\n";
  }
  file.close();
  return "data saved!\n";
}

std::string load(point tab[], int& n, std::string path)
{
  std::fstream file;
  file.open(path, std::fstream::in);
  if (file.is_open()) {
    file >> n;
    for (int i = 0; i < n; i++) {
      file >> tab[i].x >> tab[i].y;
    }
    file.close();
    return "data loaded!\n";
  }
  else return "file not found!\n";
}

int main()
{
  point A = { 1.5,-1.5 };
  point points[5];
  int n = 0;
  print(A);
  rnd(points, 5);
  for (int i=0; i < 5; i++) print(points[i]);
  std::cout << save(points, 5, "out.txt");
  std::cout << load(points, n, "out.txt");
  for (int i = 0; i < n; i++) print(points[i]);
}
```

2 Zadanie

Napisz prosty program bazodanowy. Wykorzystaj w tym celu tablicę statyczną. Należy zapewnić następujące funkcje i wykorzystać struktury danych:

1. OCENA 3:

   - wczytanie z pliku,
   - zapis do pliku,
   - wyświetlenie bazy,

   Dane odczytuj i zapisuj do pliku `*.txt`. Poszczególne dane w rekordzie oddzielaj spacjami, a poszczególne rekordy zapisuj jako nowe linie.

2. OCENA 4:

   - usuwanie wybranego elementu,
   - dodawanie elementu do bazy,
   - losowanie i wyświetlenie elementu.

3. OCENA 5:

   - wyszukiwanie po wybranym polu struktury,
   - filtrowanie po wybranym polu struktury (np. wyfiltrowanie wszystkich kobiet lub modeli samochodu),
   - sortowanie po wybranym polu struktury.

   Przykładowe struktury danych:

   - student
     - nr indeksu,
     - imię,
     - nazwisko,
     - płeć (typ *char*, m – mężczyzna, k – kobieta),
     - data urodzenia (deklaracja jako *string*),
   - samochód
     - nr rejestracyjny,
     - marka,
     - model,
     - typ,
     - rocznik,
     - moc.

W kodzie pisanych programów proszę umieszczać komentarze. Brak komentarzy uniemożliwi zdobycie maksymalnej ilości punktów za zadania!

Uwaga

Przypominam o przesłaniu programów na koniec zajęć, według wcześniej podanego wzoru. Proszę o niedołączanie plików innych niż o rozszerzeniu `*.cpp`. – mam jakieś przykładowe, ale muszę posprawdzać, jakby co to pisz to wstawię to co mam